Atitit 集合运算topic--选择 运算语言 spel

10:集合的选择

SpEL通过collection.?[condition\_expr]对集合按照特定的筛选条件进行筛选。只有符合条件的集合才会被筛选出来。

public class SpelGrammar

{

public static void main(String[]args)

{

// 创建一个ExpressionParser对象，用于解析表达式

ExpressionParser parser = new SpelExpressionParser();

List<String>list=new ArrayList<String>();

list.add("java");

list.add("PHP");

list.add("JavaScript");

Map<String,Double>map=new HashMap<String,Double>();

map.put("math", 92.8);

map.put("chinese", 98.6);

map.put("english", 92.2);

// 创建一个EvaluationContext对象，作为SpEL解析变量的上下文

EvaluationContext ctx = new StandardEvaluationContext();

// 设置两个变量

ctx.setVariable("list", list);

ctx.setVariable("myMap", map);

//长度大于7的list集合元素将被筛选出来

Expression expr=parser.parseExpression("#list.?[length()>7]");

System.out.println("符合条件的List元素是："+expr.getValue(ctx));

//key的长度小于5且value>90的map元素将被筛选出来

expr=parser.parseExpression("#myMap.?[key.length()<5&&value>90]");

System.out.println("符合条件的Map值为:"+expr.getValue(ctx));

}

}

程序对list集合、map集合通过collection.?[condition\_expr]进行了筛选，这里需要注意的是，当操作Map集合的时候，需要显式地用key引用Map Entry的key，用value引用Map Entry的value，比如咱们筛选条件是：Map key的长度需要小于5，那就是#myMap.?[key.length()<5]。如果咱们的筛选条件是：Map value的值大于90，那就是#myMap.?[value>90]。如果筛选条件需要同时满足以上两条条件就是：#myMap.?[key.length()<5&&value>90]

运行结果：

符合条件的List元素是：[JavaScript]

符合条件的Map值为:{math=92.8}

11：集合的投影

集合投影就是原集合按照一定的“投影条件”对原集合的每一个元素进行投影，把“投影出来的影子”组成新的集合，它和集合的筛选不一样，筛选是筛选出来符合条件的集合元素，但是并不组成新的集合，但是投影则需要被组成新的集合。

SpEL投影运算的语法格式：

collection.![condition\_expr],和集合筛选很像，筛选是？，是不是符合条件？符合被筛选出来，而投影则是！。

public class SpelGrammar

{

public static void main(String[]args)

{

// 创建一个ExpressionParser对象，用于解析表达式

ExpressionParser parser = new SpelExpressionParser();

List<String>list=new ArrayList<String>();

list.add("java");

list.add("PHP");

list.add("JavaScript");

// 创建一个EvaluationContext对象，作为SpEL解析变量的上下文

EvaluationContext ctx = new StandardEvaluationContext();

// 设置变量

ctx.setVariable("list", list);

//------------在SpEL中对集合进行投影-----------

//将每个集合元素进行截取，组成新的集合

Expression expr=parser.parseExpression("#list.![substring(1,3)]");

System.out.println("投影后的新集合为："+expr.getValue(ctx));

List<Person>list2=new ArrayList<Person>();

list2.add(new Person(1,"VipMao",130));

list2.add(new Person(2,"ZhuLin",105));

ctx.setVariable("mylist2", list2);

expr=parser.parseExpression("#mylist2");

System.out.println("投影前的集合为："+expr.getValue(ctx));

//投影条件是 只要name属性

expr=parser.parseExpression("#mylist2.![name]");

System.out.println("投影后的新集合为"+expr.getValue(ctx));

}

}

上面通过投影条件依次对集合的每一个元素进行了投影，然后组成了新的集合，上面也说到了投影和筛选的不同，筛选是对满足条件的集合元素进行筛选，不满足的直接略过，但是投影不行，他需要对每一个集合元素都完成投影，如果有一个集合元素，没法满足投影条件导致投影失败，程序就会报错，比如我们将第一个集合投影条件改为截取2-5个长度，但是java、php的长度都不到5，那就会报错。第二个集合用到了Person类，以下

public class Person {

private Integer id;

private String name;

private int weight;

public Person(Integer id, String name, int weight) {

this.id = id;

this.name = name;

this.weight = weight;

}

//省略所有set、get方法

public String toString(){

return "id:"+id+",name:"+name+",weight:"+weight;

}

}

运行结果：

投影后的新集合为：[av, HP, av]

投影前的集合为：[id:1,name:VipMao,weight:130, id:2,name:ZhuLin,weight:105]

投影后的新集合为[VipMao, ZhuLin]

# ognl